Relatório de pesquisa de TCC II

Lucas Voltolini

4 de Julho de 2018

Conteúdo

Esta monografia foi julgada adequada como **TRABALHO DE CONCLUSÃO DE CURSO** no Curso de Matemática — Habilitação Licenciatura e aprovada em sua forma final pela Banca Examinadora designada.

Prof: Sonia Elena Palomino Castro

Coordenadora do curso de graduação em matemática

Banca Examinadora

Prof. Melissa Weber Mendonça

Orientadora

(Universidade Federal de Santa Catarina)

Matheus Cheque Bortolan.

Prof. Matheus Cheque Bortolan

(Universidade Federal de Santa Catarina)

Fábio Jr. Margotti

Prof. Fábio Junior Margotti

(Universidade Federal de Santa Catarina)

Agradecimento

Dedico este trabalho aos meus familiares pelo pleno apoio em todas as etapas de minha vida acadêmica e por todas as intervenções e incentivos ao estudo. Agradeço em especial aos meus pais e meus avós pela educação e por abrirem mão de tantas coisas para me dar o que não tiveram.

Dedico este trabalho também aos meus amigos mais próximos. Ao Carlos, Jaqueline e Paulo agradeço por estarem sempre presentes e disponíveis para mim de forma incansável. Ao Juan e ao Norton agradeço agradeço pelo forte e constante companheirismo.

Por fim, dedico este trabalho para a minha orientadora Melissa pela paciência e dedicação colocados neste trabalho.

Abstract

In mathematics, the chosen method to get a result depends on what one wants to solve. In optimization, we tend to separate each problem in relation to the type of function we try to minimize: we may consider how many times the function is differentiable in a given region, the so-called feasible region $\Omega \subset \mathbb{R}^n$ where we're looking for a local minimum or the constraints of the problem and there is always a large number of results for each situation.

In this work, we'll study the use of augmented Lagrangian methods in functions that satisfy the necessary conditions for this purpose. An optmization problem is a problem where we want minimize a function $f\colon \mathbb{R}^n \to \mathbb{R}$, usually with some constraints c_i and c_j, which are functions from $\mathbb{R}^n$ to $\mathbb{R}$. This way, we want to find a $x \in \mathbb{R}^n$ whish is a solution to the problem

$$\text{minimize } f(x) \text{ given } c_i(x) = 0, c_j(x) \geq 0$$

where $i \in \mathcal{E}$ and $j \in \mathcal{I}$ are indexes.

Optimization problems usually require us to understand concepts like constraints, convergence and iteration. For those who are not used to the methods described in here, we preparedan appendix with some useful definitions.

Resumo

Na matemática, em geral, o método usado para se chegar em soluções depende do que se quer resolver. Na otimização, tende-se a separar cada problema em relação ao tipo de função que se tenta minimizar: seja pelas derivadas que a função tem em uma região, pelo subconjunto $\Omega \subset \mathbb{R}^n$ onde procuramos um mínimo local ou pelas restrições do problema, há sempre um grande número de teoremas para cada situação.

Neste projeto, estudaremos a resolução de problemas com restrições usando as lagrangianas aumentadas (ou o método de lagrangiana aumentada) em funções que satisfazem as condições necessárias para tanto.

Um problema é dito de otimização se desejamos minimizar uma função de $\mathbb{R}^n$ em $\mathbb{R}$, geralmente com algumas restrições c_i e c_j que são funções de $\mathbb{R}^n$ em $\mathbb{R}$. Ou seja, queremos encontrar x que é solução do problema

$$\text{minimizar } f(x) \text{ sujeito a } c_i(x) = 0, c_j(x) \geq 0$$

em que $i \in \mathcal{E}$ e $j \in \mathcal{I}$ são os índices de igualdade e de desigualdade respectivamente.

Problemas de otimização requerem o entendimento de conceitos como restrições, convergência e iteração. Para os que não estão acostumados com os métodos descritos, nós preparamos um apêndice com definições e teoremas úteis.

Capítulo

1

Fundamentos de otimização irrestrita

1.1 Princípios e definições

Seja $f: \mathbb{R}^n \to \mathbb{R}$. Considere a situação em que precisamos determinar

$$\min f(x), \text{ sujeito a } x \in \mathbb{R}^n$$

A este problema damos o nome de **problema de minimização irrestrito**.

Para contexto deste trabalho, f é sempre definida como função de $\mathbb{R}^n$ a $\mathbb{R}$, a menos que seja devidamente explicitado o contrário.

Seja a situação em que queremos encontrar $x \in \mathbb{R}^n$ que minimiza uma função f.

Definição 1.1.1. *Dizemos que $x \in \mathbb{R}^n$ é um minimizador global de $f: \mathbb{R}^n \longrightarrow \mathbb{R}$ se*

$$f(x) \leqslant f(y)$$

para todo $y \in \mathbb{R}^n$.

Em muitos casos, f é tal que é possível encontrar um minimizador global estrito para o problema:

Definição 1.1.2. *Dizemos que $x \in \mathbb{R}^n$ é um* **minimizador global estrito** *de $f: \mathbb{R}^n \longrightarrow \mathbb{R}$ se*

$$f(x) < f(y)$$

para todo $y \in \mathbb{R}^n$, $x \neq y$

Definição 1.1.3. *Dizemos que* $v = f(x)$ *é o* **valor ótimo** *de* f *se* x *é um minimizador global.*

Em casos em que a função objetiva é **convexa**, ou seja, quando para quaisquer $x, y \in \mathbb{R}^n$ vale

$$f(\alpha x + (1-\alpha)y) \leqslant \alpha f(x) + (1-\alpha)f(y), \forall \alpha \in [0,1]$$

podemos deduzir diversos teoremas que dão uma fácil aproximação do minimizador da função. Entretanto, quando f não é convexa é muito difícil encontrar um minimizador global da função e o cálculo de um minimizador local é muito mais simples em usando algoritmos que veremos adiante.

Consideremos $\mathbb{R}^n$ como um espaço vetorial no corpo dos reais:

Definição 1.1.4. *Sejam* $u, v \in \mathbb{R}^n$, α *um escalar. Uma* **norma** *em* $\mathbb{R}^n$ *é uma função* $\|\cdot\|: \mathbb{R}^n \times \mathbb{R}^n \to [0, \infty]$ *onde valem as seguintes propriedades:*

i. $\|u\| \geqslant 0$. $\|u\| = 0 \Leftrightarrow u = 0$

ii. $\|\alpha \cdot u\| = |\alpha| \cdot \|u\|$

iii. $\|u + v\| \leqslant \|u\| + \|v\|$ *(desigualdade triangular)*

Exemplo 1.1.5. *Seja* X *um espaço vetorial e seja o vetor* $u \in X$ *tal que* $u = (\xi_1, \xi_2, \cdots, \xi_n)$. *(*$\xi_j$*) são os coeficientes de* u *na base* $\{x_j\}$*). Algumas normas conhecidas são:*

Norma 1. $\|\cdot\|_1$*:*

$$\|u\|_1 = \sum_{j=1}^{n} |\xi_j|$$

Norma 2. $\|\cdot\|_2$*:*

$$\|u\|_2 = \left(\sum_{j=1}^{n} |\xi_j|^2\right)^{\frac{1}{2}}$$

Norma p. $\|\cdot\|_p$:

$$\|u\|_p = \left(\sum_{j=1}^{n} |\xi_j|^p\right)^{\frac{1}{p}}$$

Norma infinito. $\|\cdot\|_\infty$

$$\|u\|_\infty = \max\{|\xi_j|\}$$

Definição 1.1.6. *Sejam $0 < \delta \in \mathbb{R}$ e $a \in \mathbb{R}^n$. Dizemos que $B_\delta(a) = \{x \in \mathbb{R}^n, \|x - a\| < \delta\}$ é a* **bola aberta** *de raio δ e centro em a.*

Entendemos por aqui que $\mathbb{R}^n$ com a norma $\|\cdot\|_2$ é um espaço vetorial normado, ou seja, é um espaço vetorial provido de uma função norma.

Definição 1.1.7. *Dizemos que uma* **vizinhança** *de a é um conjunto que contém uma bola aberta centrada em a.*

Definição 1.1.8. *Seja uma vizinhança $\mathcal{V}_x \subset \mathbb{R}^n$. Dizemos que x é um* **minimizador local** *de $f\colon \mathbb{R}^n \longrightarrow \mathbb{R}$ se $x \in \mathcal{V}_x$ e temos:*

$$f(x) \leqslant f(y)$$

para todo $y \in \mathcal{V}_x$.

Para o minimizador local estrito vale uma afirmação análoga:

Definição 1.1.9. *Seja uma vizinhança $\mathcal{V}_x \subset \mathbb{R}^n$. Dizemos que x é um* **minimizador local estrito** *de $f\colon \mathbb{R}^n \longrightarrow \mathbb{R}$ se $x \in \mathcal{V}_x$ e temos:*

$$f(x) < f(y)$$

para todo $y \in \mathcal{V}_x$, $x \neq y$.

Definição 1.1.10. *Dizemos que um problema é* **irrestrito** *se o problema se trata de encontrar x que minimiza $f(x)$ entre todos os elementos de $\mathbb{R}^n$.*

Neste trabalho estudaremos algoritmos de otimização restrita, ou seja, que não é irrestrita. Por notação introduzimos dois conjuntos $\mathcal{E}$ e $\mathcal{I}$ de índices que são importantíssimos para o desenvolvimento teórico na otimização restrita:

Definição 1.1.11. *Dizemos que o problema de minimizar a função objetivo*

$$\min f(x)$$

é **restrito** *quando x está condicionado por funções c_i, c_j. Estas funções são chamadas de* **restrições do problema** *de minimizar f.*

Definição 1.1.12. *Considere o problema de minimizar $f(x)$ sujeito a funções $c_i(x) = 0$ e $c_j(x) \geq 0$. Dizemos que $\mathcal{E} = \{i \in \mathbb{N} | c_i(x) = 0\}$ é o conjunto de índices de igualdade e $\mathcal{I} = \{j \in \mathbb{N} | c_j(x) \geqslant 0\}$ é o conjunto de índices de desigualdade.*

Exemplo 1.1.13. *Considere $x = (x_1, x_2) \in \mathbb{R}$, $f(x_1, x_2) = x_1^2 + x_2$. Então*

$$\min f(x) \text{ sujeito a } 2x_1 - 1 \geqslant 0, x_2 - 1 = 0$$

é um problema de minimização restrito com $c_1(x) = 2x_1 - 1$ restrição de desigualdade e $c_2(x) = x_2 - 1$ restrição de igualdade.

Definição 1.1.14. *Se o problema de minimizar f é restrito e Ω pode ser identificado por:*

$$\Omega = \{x \in \mathbb{R}^n | c_i(x) = 0, c_j(x) \geqslant 0\}$$

então $c_i \colon \mathbb{R}^n \longrightarrow \mathbb{R}$, $i = 1, \ldots, l$ são as **restrições de igualdade** *e $c_j \colon \mathbb{R}^n \longrightarrow \mathbb{R}$, $j = 1, \ldots, m$ são as* **restrições de desigualdade***.*

Muitas vezes em otimização precisamos desenvolver algoritmos iterativos para facilitar o cálculo do minimizador de uma função. O ponto é que muitos métodos sugerem que usemos informações da própria função objetivo (como por exemplo as derivadas contínuas de primeira e segunda ordem da função), o que garante uma nova gama de oportunidades de atuação na otimização de funções. Muitos destes métodos serão estudados nos próximos capítulos.

Teorema 1.1.15. Teorema de Taylor simplificado *Sejam* $f\colon \mathbb{R}^n \to \mathbb{R}$ *continuamente diferenciável e* $p \in \mathbb{R}^n$. *Então*

$$f(x+p) = f(x) + \langle \nabla f(x+tp), p \rangle$$

para algum $t \in (0,1)$. *Além do mais, se* f *é duas vezes diferenciável, temos*

$$f(x+p) = f(x) + \langle \nabla f(x), p \rangle + \frac{1}{2} \langle \nabla^2 f(x + t \cdot p)p, p \rangle$$

para algum $t \in (0,1)$.

Demonstração. Disponível em [8]. □

Muitos métodos que serão vistos usam as informações de primeira e segunda derivadas da função para aproximar um minimizador local. As informações adquiridas desta forma são geralmente usadas como critério para verificar se a sequência convergiu para um minimizador ou não.

Teorema 1.1.16. (Condição necessária de primeira ordem) *Se* x *é um minimizador local de* $f\colon \mathbb{R}^n \to \mathbb{R}$ *e* f *é continuamente diferenciável em* x, *então*

$$\nabla f(x) = 0$$

Se x *satisfaz a condição necessária de primeira ordem, dizemos que* x *é um* **ponto estacionário**.

Teorema 1.1.17. (Condição necessária de segunda ordem) *Se* x *é um minimizador local de* $f\colon \mathbb{R}^n \to \mathbb{R}$ *e* f *é duas vezes diferenciável em* x, *então*

$$\langle \nabla^2 f(x)d, d \rangle \geqslant 0, \forall d \in \mathbb{R}^n$$

A existência por fim de condições suficientes nos mostra que podemos aplicar estas condições em busca de minimizadores locais da função objetivo f.

__Teorema__ **1.1.18. (Condição suficiente de primeira ordem)** *Se $f : \mathbb{R}^n \to \mathbb{R}$ é uma função duas vezes diferenciável no ponto x, satisfaz a condição necessária de segunda ordem e se x é um ponto estacionário, se*

$$\langle \nabla^2 f(x)d, d\rangle > 0, \forall d \in \mathbb{R}^n$$

então x é um minimizador local de f.

As provas para estes teoremas se encontram em [2]

Capítulo 2

Fundamentos de algoritmos para resolução de problemas de otimização

A forma geral de um problema de minimização é

$$\min f(x), \text{ sujeito a } c_i(x) = 0, c_j(x) \geq 0$$

onde $i \in \mathcal{E}$ e $i \in \mathcal{I}$. Assim, a ideia é encontrar x^* tal que x^* minimiza a função entre todos os elementos de $\mathbb{R}^n$ que respeitam as restrições. Entendemos que a otimização, para o propósito deste documento, consiste em encontrar minimizadores de funções que modelam todos os detalhes de um problema ou uma situação e esta comporta-se sempre de acordo com as condições dadas inicialmente, o que chamamos de problema determinístico.

Este trabalho se dedica a estudar os algoritmos de otimização que buscam minimizar funções reais que suportam estas condições.

Algoritmos de otimização são programas iterativos onde, a partir de uma estimativa inicial de minimizador, gera-se uma sequência de estimativas para o minimizador cada vez melhores, ou seja, cada vez mais próximas da solução verdadeira. Como veremos adiante, cada algoritmo está condicionado a fatores como robustez (a capacidade de resolver um grande número de problemas), eficiência (associado ao custo computacional envolvido em calcular o minimizador) e precisão (o quão precisa é a aproximação gerada pelo código). Note que as qualidades não são necessariamente excludentes para todos os problemas.

Quando tratamos de resolver os problemas de minimização no computador é, muitas vezes, mais fácil tratar um problema irrestrito. Na verdade, muitas vezes o custo de encontrar um ponto no conjunto viável do problema pode ser maior do que minimizar a função dentro das restrições. Estes problemas direcionam o nosso estudo para adapatar problemas restritos a problemas sem restrição de modo que este último seja equivalente ao problema original.

2.1 Métodos de busca linear

Entendemos que há duas formas principais de qualificar os algoritmos de minimização: busca linear e região de confiança. Quando nos referimos a um método de região de confiança, o que se busca é minimizar uma função modelo da função f numa região que contém x. Os métodos de região de confiança merecem a devida atenção, mas não serão tratados neste trabalho.

Já os métodos de busca linear baseiam-se na ideia de gerar uma sequência que aproxima o minimizador da função através de "passos", ou seja, para cada iteração o novo termo da sequência é dado pela seguinte fórmula geral:

$$x_{k+1} = x_k + \alpha_k p_k$$

Em que p_k é a **direção de descida** por onde o algoritmo buscará minimizar a função e α_k é o **comprimento de passo**. Dizemos que p_k é uma direção de descida se $\langle p_k, \nabla f(x)\rangle < 0$.

A escolha mais intuitiva para p_k quando queremos minimizar $f \colon \mathbb{R}^n \to \mathbb{R}$ é dada por $\nabla f = -p_k$. Por propriedade do vetor gradiente, p_k é a direção em que a função decresce mais rapidamente.

O algoritmo de busca linear que usa o vetor gradiente como direção de descida é conhecido como **Método do gradiente** e este será estudado adiante.

Exemplo 2.1.1. *Outro método bastante eficaz de minimização irrestrita é conhecido por* **método de Newton**. *Este se baseia em escolher a direção de descida através do uso (quando possível) das derivadas de segunda ordem da função f. Neste caso, a direção de descida é defi-*

nida por $(\nabla^2 f(x_k))^{-1}\nabla f(x_k) = -p_k$.

Como veremos adiante, o método de Newton é extremamente útil quando consideramos que a aproximação inicial x *está relativamente próxima ao minimizador* x^* *da função.*

2.1.1 Comprimento de passo e condições de Wolfe

Quando usamos um algoritmo de busca linear, primeiro calculamos a direção e, logo em seguida, o comprimento de passo. A eficiência deste algoritmo depende da eficiência na hora de escolher tanto a direção quanto α_k. [1]

Quanto ao p_k, boa parte das direções de descida são dadas na forma:

$$B_k^{-1}\nabla f(x_k) = -p_k$$

onde B é uma matrix simétrica e não singular. Temos $B_k = I$ no método do gradiente e $B_k = \nabla^2 f(x_k)$ no método de Newton. Em métodos de Quase-Newton, a matriz B é dada por uma aproximação da matriz Hessiana. Para o leitor interessado, todos estes métodos estão descritos com detalhes em [1].

Dado o momento em que decidimos a direção de descida, precisamos calcular de maneira rápida o comprimento de passo α_k que garante o funcionamento do método. Seja a função:

$$\phi(\alpha) = f(x_k + \alpha p_k)$$

Se conseguirmos minimizar ϕ então teremos um bom comprimento de passo para a função f aplicada no ponto x_k. Calcular α é caro do ponto de vista computacional e, por esta razão, buscamos maneiras eficientes de encontrar um bom comprimento de passo.

As **condições de Wolfe** são um conjunto de soluções extremamente populares nos métodos computacionais para encontrar um comprimento de passo de forma barata e segura ao exigir que α consiga "diminuir o suficiente"a função objetivo. Queremos que α seja tal que para $c \in (0, 1)$:

$$f(x_k + \alpha p_k) \leq f(x_k) + c\langle \alpha \nabla f(x), p_k \rangle$$

Se α é tal que satisfaz a primeira condição, introduzimos a segunda condição de Wolfe, conhecida por **condição de curvatura**:

$$\langle \nabla f(x_k + \alpha \cdot p_k), p_k \rangle \geq \langle c' \nabla f(x_k), p_k \rangle$$

em que $c < c' < 1$. Note que a condição de curvatura acaba comparando o gradiente de $\phi(\alpha)$ com $c' \cdot \nabla\phi(0)$ e garantindo que α minimiza a função na direção p_k.

No entanto, nem sempre teremos α que satisfaz as condições de Wolfe e que, ao mesmo tempo, minimiza a função ϕ. Neste caso, dizemos que α satisfaz a **condição forte de Wolfe** se:

1. $f(x_k + \alpha \cdot p_k) \leq f(x_k) + \langle c\alpha \cdot \nabla f(x), p_k \rangle$

2. $|\langle \nabla f(x_k + \alpha \cdot p_k), p_k \rangle| \geq |\langle c' \nabla f(x_k), p_k \rangle|$

para $0 < c < c' < 1$.

2.1.2 Backtracking

Muitas vezes pode ser mais barato computacionalmente escolher um comprimento de passo com algum método de "força bruta", ou seja, que dá sugestões de possíveis valores de α para cada iteração. Considere o seguinte algoritmo:

1. Dados $\alpha_0 > 0, c_0, c_1 \in (0, 1)$;
2. Para $k = 0, 1, 2, \cdots$:
   - Se $\phi(\alpha_k) < \phi(0) + \langle c_0 \alpha_k \nabla f(x), p \rangle$, pare;
   - $\alpha_{k+1} = c_1 \cdot \alpha_k$

A este famoso método damos o nome de **Backtracking**. Trata-se de uma forma simples de calcular o novo valor α_k com amplo uso em computação.

Exemplo 2.1.2. *Considere o seguinte algoritmo do método do gradiente:*

1. *Escolha* $x_0 \in \mathbb{R}^n, \alpha = 0.9$ *;*

2. *Para* $k = 0, 1, 2, \cdots$*:*

   tome $s = \alpha$ *e* $x_{k+1} = x_k - \nabla f(x_k)$

   Enquanto $f(x_{k+1}) > f(x_k)$*;*

   $x_{k+1} = x_k - s \cdot \nabla f(x_k)$

   $s = \alpha \cdot s$

Este algoritmo representa uma forma de otimizar a função $f(x)$ *através de Backtracking que reduz o comprimento do passo em 10% a cada nova iteração.*

2.2 Otimização restrita e multiplicadores de Lagrange

Definição 2.2.1. *Dizemos que* $\Omega \subset \mathbb{R}^n$ *é o conjunto viável do problema de minimizar a função objetivo se o problema está condicionado por funções* $c_i \colon \mathbb{R}^n \to \mathbb{R}$ *e* $c_j \colon \mathbb{R}^n \to \mathbb{R}$ *que são conhecidas por restrições de igualdade e de desigualdade.*

$$f \colon \mathbb{R}^n \longrightarrow \mathbb{R}$$

Até agora vimos que os problemas de minimização podem conter restrições de igualdade e de desigualdade que definem o conjunto viável da função. Considere um problema de minimização restrito

$$\min f(x), \text{ sujeito a } c_i(x) = 0, c_j(x) \geq 0$$

onde $i \in \mathcal{E}$ e $j \in \mathcal{I}$.

Definição 2.2.2. *Dizemos que* $A(x)$ *é o* **conjunto ativo** *de* x *se*

$$A(x) = \mathcal{E} \cup \{j \in \mathcal{I}, c_j(x) = 0\}$$

Ou seja, $A(x)$ é um conjunto de índices. Se $c_j(x) = 0$ com $j \in \mathcal{I}$, então c_j é uma restrição ativa de x. Mais do que isto, temos:

Definição 2.2.3. *Considere o problema de minimizar $f(x)$ restrito. Dizemos que x é* **LICQ** *se $\nabla c_i(x)$ existem e são linearmente independentes para $i \in A(x)$.*

O termo "LICQ", abreviação para "linear independent constraint qualification"corresponde a uma caracterização das restrições do problema de minimizar $f(x)$ restrito. Por fim, gostaríamos de definir as condições de otimalidade para o problema restrito:

Definição 2.2.4. *Sejam $x \in \Omega$, $\lambda_{i_1}, \lambda_{i_2}, \ldots, \lambda_{i_{|A(x)|}}$ escalares tais que $i_1, i_2, \ldots, i_{|A(x)|} \in A(x)$. Dizemos que $\mathcal{L}(x, \lambda)$ é a* **função lagrangiana** *de f se*

$$\mathcal{L}(x, \lambda) = f(x) - \lambda_i c_i(x)$$

para $i \in A(x)$.

Definição 2.2.5. *Seja x minimizador local de um problema de minimização restrito com x LICQ. Dizemos que o problema satisfaz as condições de* **Karush-Kuhn-Tucker** *se existe um vetor λ chamado de* **multiplicador de Lagrange** *tal que:*

1. $\nabla \mathcal{L}(x, \lambda) = 0$;

2. $c_i(x) = 0$ *se c_i é condição de igualdade;*

3. $c_j(x) \geq 0$ *se c_j é condição de desigualdade;*

4. $\lambda_i \geq 0$;

5. $\lambda_i c_i(x) = 0, \forall i \in A(x)$.

em que $\mathcal{L}$ é a função **lagrangiana** *do problema de minimizar f com restrições.*

A função lagrangiana e as **condições de KKT** são extremamente úteis para calcular a convergência de métodos de otimização restrita. Adiante verificaremos a importância destas novas definições para o método de lagrangianas aumentadas.

Capítulo 3

Ferramentas para construção dos algoritmos

3.1 Python

Para a implementação dos algoritmos em problemas de otimização decidimos usar uma linguagem de programação comum para o cálculo destes problemas, o Python.

Python é uma linguagem de programação interativa de alto nível com multiparadigma (incluindo o paradigma funcional, orientação a objetos e programação estruturada). Atualmente Python vem acompanhada de diversas bibliotecas e módulos que expandiram seu uso para diversos problemas. Seu desenvolvimento é livre e seu uso na construção de programas não requer licença de direitos autorais.

Criado por Guido Van Rossum em 1989 e lançado em 1991, Python é extremamente simples e dinâmica. Seu desenvolvimento fundamentou-se na linguagem ABC, utilizada no Instituto de Pesquisa Nacional para Matemática e Ciência da Computação (CWI), onde Rossum atuava durante a década de 80. Desde o seu lançamento, a linguagem tem sido aprimorada por meio das comunidades contribuintes, e atualmente está em sua versão 3.6.5. A versão inicial já possuía classes com herança, tratamento de exceções, funções e os tipos de dado "list", "dict", e "str", respectivamente listas, dicionários e "strings". Em 1994 foi lançada uma nova versão do código, chamado de 1.0. Nesta versão foram inseridas ferramentas para programação funcional como a função "lambda", "map", "filter"e "reduce". O lançamento da versão 1.0 coincidiu com o surgimento do primeiro fórum de discussão do Python que coordena ainda hoje as diretivas do desenvolvimento de código. Entre a primeira versão até a versão 2.0 apresentada no ano de 2000, a linguagem Python foi aperfeiçoada com o incremento de várias funcionalidades de outras linguagens.

São muitos os fatores que popularizaram a linguagem Python, a começar pela facilidade de se traduzir de forma algorítmica. Isso acontece graças ao fato de que as palavras-chave utilizadas nesta linguagem objetivam a estruturação dos programas. Por fim, a linguagem Python é versátil e abrangente podendo ser empregada em diversos tipos de tarefas.

Hoje, com 62 anos, Rossum ainda supervisiona o desenvolvimento da linguagem e o avanço da comunidade mundial dedicada ao seu invento.

3.1.1 NumPy.Linalg

NumPy é um projeto voluntário de código aberto para o uso em computação científica. Em especial, o programa oferece suporte para uso de matrizes e cálculo de funções que operam nestas matrizes.

A ideia surgiu na comunidade Python quando a linguagem passou a ser usada em computação científica. Em partes baseando-se no Numeric e Numarray, os colaboradores criaram o Numpy: uma biblioteca destinada a oferecer o suporte a cálculos avançados de forma semelhante ao MATLAB implementado em Python.

Atualmente o desenvolvimento do Numpy se dá através da criação de pacotes dentro da própria biblioteca com diversas novas funções, além da contribuição da comunidade em projetos parceiros como o Scipy, Matplotlib (representação gráfica) e Numba (otimização).

Neste projeto usamos o pacote conhecido como "linalg", que tem como função facilitar a programação científica com elementos de álgebra linear.

3.2 Método do gradiente

O primeiro algoritmo para a minimização irrestrita de funções é um dos mais antigos métodos de otimização para funções de $\mathbb{R}^n$ e baseia-se na ideia de usar a direção de descida da função como o oposto do gradiente, ou seja, dado um ponto inicial e f continuamente diferenciável, para cada iteração tomamos $\nabla f(x_k) = -p_k$.

Assim, o método gera uma sequência $x_{k+1} = x_k + \alpha_k p_k$ em que α é o comprimento de passo.

Algoritmo:

1. Escolha $x_0 \in \mathbb{R}^n, c \in (0,1)$;

2. Para $k = 0, 1, 2, \cdots$:

    tome $s = c\nabla f(x_k)$ e $x_{k+1} = x_k - \nabla f(x_k)$

    Enquanto $f(x_{k+1}) > f(x_k)$;

    $x_{k+1} = x_k - s$

    $s = cs$

    $k = k + 1$

O método do gradiente é em geral um método lento e não será abordado com grande ênfase neste documento. Para o leitor interessado, recomendamos a leitura do livro Numerical Optmization [1].

3.3 Método de Newton

O método de Newton é um algoritmo usado para encontrar raízes de uma função real. Pelas condições de otimalidade, gostaríamos de encontrar x que seja ponto estacionário e, portante, buscar raízes para o gradiente de f..

Seja $f : \mathbb{R}^n \to \mathbb{R}$ diferenciável. Nosso problema consiste em encontrar um mínimo de f. De acordo com as condições de otimalidade, precisamos encontrar um ponto $x^* \in \mathbb{R}^n$ tal que

$$\nabla f(x^*) = 0$$

e

$$\nabla^2 f(x^*) > 0$$

Usando a expansão em série de Taylor de f em torno do ponto x_k podemos aproximar a função objetivo por um modelo quadrático $m_k(d_k)$, em que

$$f(x_k + d_k) \approx f(x_k) + \langle \nabla f(x_k), d_k \rangle + \frac{1}{2} \langle d_k \nabla^2 f(x_k), d_k \rangle + \mathcal{O}(\|d_k\|^3)$$

Assim, temos o modelo quadrático que aproxima f em torno de x_k

$$m_k(d_k) = f(x_k) + \langle \nabla f(x_k), d_k \rangle + \frac{1}{2} \langle d_k \nabla^2 f(x_k), d_k \rangle$$

Visto que nosso objetivo é minimizar f, vamos tentar encontrar d de forma a minimizar $m_k(d_k)$. Assumindo que $\nabla^2 f(x_k) > 0$ e derivando $\nabla m_k(d_k) = 0$ em relação a d_k obtemos

$$\nabla m_k(d_k) = \nabla f(x_k) + \nabla^2 f(x_k) d_k = 0$$

$$\nabla^2 f(x_k) d_k = -\nabla f(x_k)$$

$$d_k = -(\nabla^2 f(x_k))^{-1} \nabla f(x_k)$$

em que $d_k = -(\nabla^2 f(x_k))^{-1} \nabla f(x_k)$ é a direção de Newton.

Algoritmo: Dada a necessidade de aplicar o método, propomos o método de Newton com Backtracking para minimizar $f(x)$ irresttritamente:

1. Escolha $x_0, c \in (0, 1)$;
2. Para $k = 0, 1, 2, \cdots$:

    tome $s = c\nabla f(x, k)$ e $x_{k+1} = x_k - \nabla^2 f(x_k)^{-1} \nabla f(x_k)$

    Enquanto $f(x_{k+1}) > f(x_k)$;

        $x_{k+1} = x_k - s$

        $s = cs$

    $k = k + 1$

3.3.1 Convergência do Método de Newton

Teorema 3.3.1 (Convergência local do Método de Newton)**.** *Suponha que f é duas vezes diferenciável e que a matriz Hessiana $\nabla^2 f(x)$ é Lipschitz contínua em uma vizinha aberta $\mathcal{V}$ de x^* satisfazendo as condições suficientes de segunda ordem. Considere a iteração do Método de Newton dada por $x_{k+1} = x_k + \alpha_k d_k$. Então,*

1. *Se o ponto inicial x_0 está suficientemente próximo de x^*, a sequência $(x_k)_{k\in N}$ gerada pelo método de Newton converge para x^*;*

2. *x_k converge quadraticamente para x^*;*

3. *A sequência $(\|\nabla f(x_k)\|)_{k\in N}$ converge quadraticamente para zero.*

Demonstração. Dada a direção de Newton,

$$d_k = -(\nabla^2 f(x_k))^{-1}\nabla f(x_k)$$

e a condição de otimalidade $\nabla f(x^*) = 0$, temos

$$x_k + d_k - x^* = x_k - x^* - \nabla^2 f(x_k)^{-1}\nabla f(x_k)$$

$$x_k + d_k - x^* = \nabla^2 f(x_k)^{-1}[\nabla^2 f(x_k)(x_k - x^*) - (\nabla f(x_k) - \nabla f(x^*))]$$

Pelo Teorema de Taylor, temos

$$\nabla f(x_k) - \nabla f(x^*) = \int_0^1 \nabla^2 f(x_k + t(x^* - x_k))(x_k - x^*)dt$$

Para L constante (de Lipschitz) e $\nabla^2 f(x^*)$ inversível, basta k suficientemente grande:

$$\begin{aligned}\|\nabla^2 f(x_k)(x_k - x^*) - (\nabla f(x_k) - \nabla f(x^*))\| &= \left\| \int_0^1 \left(\nabla^2 f(x_k) - \nabla^2 f(x_k + t(x^* - x_k))\right)(x_k - x^*)\, dt \right\| \\ &\leqslant \int_0^1 \|\nabla^2 f(x_k) - \nabla^2 f(x_k + t(x^* - x_k)\|\|(x_k - x^*)\|\, dt \\ &\leqslant \int_0^1 Lt\|x_k - x^*\|^2\, dt\end{aligned}$$

$$= \frac{1}{2}L\|x_k - x^*\|^2$$

Assim, segue

$$\|x_k + d_k - x^*\| \leqslant L\|\nabla^2 f(x^*)^{-1}\| \cdot \|x_k - x^*\|^2 = \bar{L}\|x_k - x^*\|^2$$

Convergência:

De fato, como $x_{k+1} - x_k = -(\nabla^2 f(x_k))^{-1}\nabla f(x_k)$ e $\nabla f(x_k) + (\nabla^2 f(x_k) d_k = 0$, temos

$$\|\nabla f(x_{k+1})\| = \|\nabla f(x_{k+1}) - \nabla f(x_k) - \nabla^2 f(x_k) d_k\|$$

$$= \left\| \int_0^1 \nabla^2 f(x_k + t d_k)(x_{k+1} - x_k)\, dt - \nabla^2 f(x_k) d_k \right\|$$

que podemos facilmente estimar por

$$\|\nabla f(x_{k+1})\| \leqslant \int_0^1 \|\nabla^2 f(x_k + t d_k) - \nabla^2 f(x_k)\| \cdot \|d_k\|\, dt$$

onde, por teorema:

$$\|\nabla f(x_{k+1})\| \leqslant \frac{1}{2}L\|d_k\|^2$$

$$\leqslant \frac{1}{2}L\|\nabla^2 f(x_k)^{-1}\|^2 \|\nabla f(x_k)\|^2$$

$$\leqslant 2L\|\nabla^2 f(x^*)^{-1}\|^2 \|\nabla f(x_k)\|^2$$

$$\Rightarrow \|\nabla f(x_{k+1})\| \leqslant 2L\|\nabla^2 f(x^*)^{-1}\|^2 \|\nabla f(x_k)\|^2$$

Onde obtemos um ponto estacionário para o problema. □

Capítulo

4

Lagrangianas aumentadas e penalidade

4.1 Método de penalidade

Seja $f\colon \mathbb{R}^n \to \mathbb{R}$ uma função real. Seja $\mathcal{E}$ o conjunto de todas as restrições de igualdade c_i do problema:

$$\mathbf{min} f(x)$$

Sujeito a $i \in \mathcal{E}$

Definição 4.1.1. *A função de penalidade quadrática definida para o problema de minimizar $f(x)$ sujeita a $c_i(x) = 0, i = 0, 1, \cdots, n$ é dada por:*

$$Q(x, \mu) = f(x) + \frac{1}{2 \cdot \mu} \sum_{c_i \in \mathcal{E}} c_i(x)^2$$

Note que ao tentar minimizar $Q(x, \mu)$, estamos minimizando a função $f(x)$ em todo o conjunto viável do problema, isto é: se $x \in \Omega$, então minimizar $Q(x, \mu)$ significa minimizar $f(x)$. Caso x não respeite as restrições, aplicamos uma **penalidade** ao valor de Q que depende de uma constante μ. Assim sendo, o problema de minimizar $Q(x, \mu)$ torna-se um problema irrestrito e pode ser resolvido com algum dos métodos já estudados.

Algoritmo:

1. Escolha $\mu_0 > 0, \tau > 0, x_0$;

2. Para $k = 0, 1, 2, \cdots$:

   - Minimize a função $Q(x_k, \mu_k)$
   - Se $\nabla Q(x_k, \mu) < \tau$, pare.
   - Escolha um novo parâmetro $\mu_{k+1} > \mu_k$
   - Escolha um novo x_{k+1}
   - $k = k + 1$

Definição 4.1.2. *Seja $\mathcal{I}$ o conjunto de todas as restrições de desigualdade do problema*

$$\min f(x)$$

sujeito a $c_i(x) = 0$, $c_j(x) \geq 0$. Dizemos que a função de penalidade quadrática definida para o problema é dada por:

$$Q(x, \mu) = f(x) + \frac{1}{2\mu} \cdot \sum_{c_i \in \mathcal{E}} c_i(x)^2 + \frac{1}{2\mu} \cdot \sum_{c_j \in \mathcal{I}} \left([c_j(x)]^-\right)^2$$

onde $[c_j(x)^2]^- = \max\{0, c_j(x)\}$

Assim as iterações são feitas em x e no parâmetro μ e aconvergência do método ocorre por culpa da penalidade aplicada em pontos do $\mathbb{R}^n$ que não seguem as restrições.

4.1.1 Convergência do método de penalidade

Teorema 4.1.3. *Considere $\{x^k\}$ uma sequência de minimizadores encontrados no método de penalidade quando $\mu_k \to \infty$. Suponha também que cada x^k é um minimizador global de $Q(x, \mu)$. Então qualquer ponto de acumulação de $\{x^k\}$ é um minimizador de $f(x)$.*

Demonstração. Seja $\overline{x}$ que saitsfaz as hipóteses acima. Assim:

$$f(x^k) + \frac{\mu_k}{2} \cdot \sum c_i^2(x^k) \leqslant f(\overline{x}) + \frac{\mu_k}{2} \cdot \sum c_i^2(\overline{x}) = f(\overline{x})$$

para qualquer $x^k \in \{x^k\}$. Como $f(\overline{x}) + \frac{\mu_k}{2} \cdot \sum c_i^2(\overline{x}) = f(\overline{x})$, então:

$$\sum c_i^2(x^k) \leqslant \frac{2}{\mu_k} \cdot \Big(f(\overline{x}) - f(x^k)\Big)$$

Seja x^* ponto de acumulação de alguma subsequência de $\{x^k\}$, temos:

$$\sum c_i^2(x^*) = \lim_{k\to\infty} \sum c_i^2(x^k) \leqslant \lim_{k\to\infty} \frac{2}{\mu_k} \cdot \Big(f(\overline{x}) - f(x^k)\Big)$$

Como $\mu_k \to \infty$ para $k \to \infty$, então

$$\lim_{k\to\infty} \frac{2}{\mu_k} \cdot \Big(f(\overline{x}) - f(x^k)\Big) = 0 \Rightarrow \sum c_i^2(x^*) = 0$$

o que significa que x^* respeita as condições de restrição e é, portanto, um ponto do conjunto viável. Assim sendo, continuamos com:

$$f(x^*) \leqslant f(x^*) + \lim_{k\to\infty} \frac{\mu_k}{2} \cdot \sum c_i^2(x^k) \leqslant f(\overline{x})$$

E como $f(\overline{x})$ é um ponto de mínimo, então x^* também é minimizador global da função. □

4.2 Lagrangianas aumentadas

O uso de algoritmos de penalidade pode gerar sequências extremamente difíceis de calcular devido ao mal condicionamento da penalidade μ relacionada com o cálculo das restrições e das derivadas de primeira e segunda ordem envolvidas. Para evitar o surgimento destes problemas recorremos, em geral, ao uso de **multiplicadores de Lagrange** para controlar o comportamento da função fora do conjunto viável.

O próximo algoritmo para a minimização de funções restritas propõe uma tranformação na função original para que seja feita a resolução de um problema irrestrito. Considere a função lagrangiana com apenas uma restrição de igualdade dada por:

$$\mathcal{L}(x, \lambda_1) = f(x) - \lambda \cdot c(x)$$

Onde, pelas condições de KKT, encontramos pontos estacionários em:

$$\nabla\mathcal{L}(x, \lambda) = \nabla f(x) - \langle \lambda \cdot \nabla c(x) \rangle = 0$$

Dizemos que $\lambda_i \geq 0$ que resolve a equação acima é um multiplicador de Lagrange.

A Lagrangiana aumentada é um método que usa as propriedades da função lagrangiana para calcular com maior eficácia as novas iterações do algoritmo já que a lagrangiana estima um valor exato λ que minimiza a função sem a necessidade de tornar μ um escalar muito pequeno. A fórmula geral é dada por:

$$\mathcal{L}(x, \lambda, \mu) = f(x) - \sum \lambda_i \cdot c_i(x) + \frac{1}{2 \cdot \mu} \cdot \sum c_j^2(x)$$

em que temos por objetivo usar o valor real de λ_i em vez de uma estimativa para μ que decresce a cada iteração.

Algoritmo:

1. Escolha $\mu_0 > 0, \tau > 0, x_0^0, \lambda_0$;

2. Para $k = 0, 1, 2, \cdots$:

   - Encontre x_k que minimiza a função $\mathcal{L}(x_k^s, \lambda^k, \mu_k)$
   - Se x_k minimiza $f(x_k)$, pare;
   - Tome um novo parâmetro $\lambda^{k+1} = \lambda^k - \dfrac{c(x)}{\mu_k}$;
   - Escolha $\mu_{k+1} > \mu_k$;
   - Tome $x_{k+1}^s = x_0^{s+1}$ como novo ponto inicial.

Capítulo 5

Conclusão

Neste trabalho analisamos dois dos métodos de otimização restrita usados na minimização de funções reais. Estudamos também seu comportamento e convergência, além das implicações que eles trazem para a teoria. Nosso foco foi o estudo da teoria de otimização restrita e dos algoritmos de busca linear usados atualmente.

Estudamos no primeiro capítulo conceitos básicos de otimização restrita que são fundamentais para qualquer desenvolvimento no assunto. O conceito de algoritmo e restrições

No segundo capítulo comentamos sobre a linguagem de programação usada para a realização deste trabalho: Python. Extremamente dinâmica e acessível mesmo para pessoas com pouca experiência em programação, Python é a linguagem considerada "universal"de otimização e inteligência artificial.

No último capítulo analisamos e construímos os algoritmos que foram objetos de estudo deste documento a partir de uma visão teórica sobre seu funcionamento.

Capítulo 6

Bibliografia

Capítulo 7

Apêndice

7.1 Definições

Definição 7.1.1. *Seja F um corpo e X um conjunto não-vazio. Dizemos que X é um* **espaço vetorial sobre** *F se temos definidas duas operações $+\colon X \times X \to X$ e $\cdot\colon F \times X \to X$, chamadas respectivamente de* **adição** *e* **multiplicação por escalar***, para as quais são válidos as seguintes propriedades:*

- **Para a adição:**

  A1. *$u + v = v + u$ para todos $u, v \in X$ (comutatividade);*

  A2. *$u + (v + w) = (u + v) + w$ para todos $u, v, w \in X$ (associatividade);*

  A3. *existe um elemento $0 \in X$ tal que $u + 0 = u$ para todo $u \in X$ (elemento neutro);*

  A4. *para cada $u \in X$ existe $v \in X$ tal que $u + v = 0$, denotado por $v = -u$ (elemento oposto).*

- **Para a multiplicação por escalares:**

  M1. *existe um elemento $1 \in X$ tal que para todo $u \in X$, $1 \cdot u = u$;*

  M2. *para todo $c_1, c_2 \in F$ e para todo $u \in X$, vale $(c_1 c_2) \cdot u = c_1(c_2 \cdot u)$;*

D1. *para todo* $c \in F$ *e para todo* $u, v \in X$ *vale* $c \cdot (u + v) = c \cdot u + c \cdot v$*;*

D2. *para todo* $c_1, c_2 \in F$ *e para todo* $u \in X$*, vale que* $(c_1 + c_2) \cdot u = c_1 \cdot u + c_2 \cdot u$*.*

Cada elemento de um espaço vetorial X é chamado de **vetor**. O plano euclidiano $\mathbb{R}^2$ e o espaço euclidiano $\mathbb{R}^3$ são espaços vetoriais sobre $\mathbb{R}$, quando munidos da adição e multiplicação por escalar usuais, definidas: em $\mathbb{R}^2$ por

A. $u + v = (x_1, y_1) + (x_2, y_2) = (x_1 + x_2, y_1 + y_2)$

M. $c \cdot u = c \cdot (x, y) = (c \cdot x, c \cdot y)$

e em $\mathbb{R}^3$ por

A. $u + v = (x_1, y_1, z_1) + (x_2, y_2, z_2) = (x_1 + x_2, y_1 + y_2, z_1 + z_2)$

M. $c \cdot u = c \cdot (x, y, z) = (c \cdot x, c \cdot y, c \cdot z)$

Quando não houver confusão, o corpo F será omitido, e diremos simplesmente que X é um espaço vetorial. No decorrer deste relatório, a não ser que especificado de outra maneira, consideraremos F como sendo o corpo dos números reais $\mathbb{R}$.

A palavra "iteração"existe em muitos contextos mas, para o caso de programação, dizemos que um método é iterativo quando repetimos determinado procedimento um número de vezes até que consigamos o resultado desejado. Como exemplo, suponha que você tem listado todos os possíveis nomes de uma pessoa em uma lista finita e deseja descobrir o seu nome real. Perguntar a ela todos os nomes da lista até que ela confirme é um método iterativo para descobrir o nome de alguém. Deste modo, há vários exemplos de métodos iterativos com uso no cotidiano.

Em otimização, nosso objetivo é encontrar um ponto x^* tal que, aplicada a função neste ponto, o resultado seja o menor possível dadas algumas condições.

Seja $f\colon \mathbb{R}^n \longrightarrow \mathbb{R}$. O problema de minimizar a função f se trata, no geral, de descobrir:

$$\min f(x)$$

ou seja, encontrar $x \in \mathbb{R}^n$ tal que o valor de $f(x)$ seja mínimo.

Para contexto deste documento, $f(x)$ é sempre definida como função entre $\mathbb{R}^n \to \mathbb{R}$, a menos que seja devidamente explicitado o contrário.

Seja a situação em que queremos descobrir um $x \in \mathbb{R}^n$ que minimiza uma função f.

Definição 7.1.2. *Dizemos que $x \in \mathbb{R}^n$ é um minimizador global de $f\colon D \subset \mathbb{R}^n \longrightarrow \mathbb{R}$ se*

$$f(x) \leqslant f(y)$$

para todo $y \in D$.

Em muitos casos, f é tal que é possível encontrar um minimizador estrito para o problema:

Definição 7.1.3. *Dizemos que $x \in \mathbb{R}^n$ é um* **minimizador global estrito** *de $f\colon D \subset \mathbb{R}^n \longrightarrow \mathbb{R}$ se*

$$f(x) < f(y)$$

para todo $x \neq y \in D$.

Definição 7.1.4. *Dizemos que $v = f(x)$ é o* **valor ótimo** *de f se x é um minimizador global.*

Encontrar o minimizador global de um problema não é, no geral, uma tarefa fácil. Muitos métodos e algoritmos já desenvolvidos trabalham no intuito de encontrar pontos de mínimo da função localmente, isto é, em uma região próxima de uma aproximação inicial.

Definição 7.1.5. *Dizemos que $\mathbb{R}^n$ é o conjunto viável do problema de minimizar a função objetivo se a função é definida como:*

$$f\colon D \subset \mathbb{R}^n \longrightarrow \mathbb{R}$$

Definição 7.1.6. *Uma* **norma** *em X é uma função $\|\cdot\|\colon X \times X \Rightarrow \mathbb{R}$ onde valem as seguintes propriedades:*

i. $\|u\| \geqslant 0$. $\|u\| = 0 \Leftrightarrow u = 0$

ii. $\|\alpha \cdot u\| = |\alpha| \cdot \|u\|$

iii. $\|u+v\| \leqslant \|u\| + \|v\|$ *(desigualdade triangular)*

Exemplo 7.1.7. *Seja o vetor* $u \in X$ *tal que* $u = (\xi_1, \xi_2, \ldots, \xi_n)$. *(*$\xi_j$*) são os coeficientes de* u *na base* $\{x_j\}$*). Algumas normas conhecidas são:*

Norma 1. $\|\cdot\|_1$*:*

$$\|u\|_1 = \sum_{j=1}^{n} |\xi_j|$$

Norma 2. $\|\cdot\|_2$*:*

$$\|u\|_2 = \left(\sum_{j=1}^{n} |\xi_j|^2\right)^{\dfrac{1}{2}}$$

Norma p. $\|\cdot\|_p$*:*

$$\|u\|_p = \left(\sum_{j=1}^{n} |\xi_j|^p\right)^{\dfrac{1}{p}}$$

Norma infinito. $\|\cdot\|_\infty$*, quando* $p \to \infty$*:*

$$\|u\|_\infty = \max\{|\xi_j|\}$$

Definição 7.1.8. *Seja* $0 < \delta \in \mathbb{R}$ *e* $a \in \mathbb{R}^n$. *Dizemos que* $B_\delta(a) = \{x, \|x - a\| < \delta\}$ *é a* **bola aberta** *de raio* δ *e centro em* a

Definição 7.1.9. *Dizemos que uma* **vizinhança** *de* a *é um conjunto que contém uma bola centrada em* a.

Definição 7.1.10. *Seja uma vizinhança* $\mathcal{V}_x \subset \mathbb{R}^n$*, dizemos que* x *é um* **minimizador local** *de* $f\colon D \subset \mathbb{R}^n \longrightarrow \mathbb{R}$ *se* $x \in \mathcal{V}_x$ *e temos:*

$$f(x) \leqslant f(y)$$

para todo $y \in \mathcal{V}_x$.

Definição 7.1.11. *Seja uma vizinhança* $\mathcal{V}_x \subset \mathbb{R}^n$*, dizemos que* x *é um* **minimizador local estrito** *de* $f\colon D \subset \mathbb{R}^n \longrightarrow \mathbb{R}$ *se* $x \in \mathcal{V}_x$ *e temos:*

$$f(x) < f(y)$$

para todo $x \neq y \in \mathcal{V}_x$.

Definição 7.1.12. *Dizemos que um problema é* **irrestrito** *se minimizamos* $f\colon \mathbb{R}^n \to \mathbb{R}$ *com* $D = \mathbb{R}^n$.

Definição 7.1.13. *Dizemos que o problema de minimizar a função objetivo*

$$\min f(x)$$

é **restrito** *quando* x *está condicionado por funções* c_i, c_j. *Estas funções são chamadas de* **restrições do problema** *de minimizar* f.

Definição 7.1.14. *Se o problema de minimizar* f *é restrito e* D *pode ser identificado por:*

$$D = \{x \in \mathbb{R}^n | c_i(x) = 0, c_j(x) \leqslant 0\}$$

então $c_i\colon \mathbb{R}^n \longrightarrow \mathbb{R}$, $i = 1, \ldots, l$ *são as* **restrições de igualdade** *e* $c_j\colon \mathbb{R}^n \longrightarrow \mathbb{R}$, $j = 1, \ldots, m$ *são as* **restrições de desigualdade**.

Definição 7.1.15. *Se* $f\colon D \subset \mathbb{R}^n \longrightarrow \mathbb{R}$ *é diferenciável numa vizinhança* $\mathcal{V}_x$ *e* $\nabla f(x) = 0$, *então dizemos que* x *é um* **ponto estacionário**.

Definição 7.1.16. *Seja* c_j *uma restrição de desigualdade.* c_j *é dita ativa em* x *se, para* x *ponto do conjunto viável, então* $c_j(x) = 0$.

Definição 7.1.17. *Uma sequência é uma função com domínio em* $\mathbb{N}$.

Definição 7.1.18. *Uma* **sequência** *real é uma função* $f\colon \mathbb{N} \Rightarrow \mathbb{R}^n$.

f *pode ser representada por* $\{x_i\}$, *onde* x_i *é chamado o* **elemento da sequência** *e o conjunto imagem é o conjunto dos* **valores** *da sequência.*

Definição 7.1.19. *Uma sequência* $\{x_n\}$ *é dita* **convergente** *com limite* l *se, dado um* $\mathcal{E} \in \mathbb{R}$, $\mathcal{E} > 0$, *exista um* $n_0 \in \mathbb{N}$ *tal que para todo* $n \geqslant n_0$ *temos:*

$$\|l - x_n\| < \mathcal{E}$$

Uma sequência é dita **divergente** *se ela não for convergente.*

Definição 7.1.20. *Seja $\{x_n\}$ uma sequência. Dizemos que $\{x_n\}$ é* **limitada superiormente** *se existir um $\gamma \in \mathbb{R}$ tal que:*

$$\gamma \geq x_i$$

Para todo $i \in \mathbb{N}$.

Dizemos também que γ é um **limitante superior** *ou uma* **cota superior** *de $\{x_n\}$.*

Definição 7.1.21. *Seja $\{x_n\}$ uma sequência. Dizemos que $\{x_n\}$ é* **limitada inferiormente** *se existir um $\gamma \in \mathbb{R}$ tal que:*

$$\gamma \leq x_i$$

para todo $i \in \mathbb{N}$. Dizemos também que γ é um **limitante inferior** *ou uma* **cota inferior** *de $\{x_n\}$.*

Definição 7.1.22. *Seja $\{x_n\}$ uma sequência real. Considere a sequência $\{n_k\}$ onde $n_1 < n_2 < \cdots$. A sequência $\{x_{n_i}\}$ dada pelos elementos $\{x_{n_1}, x_{n_2}, \cdots\}$ é chamada de* **subsequência** *de $\{x_n\}$.*

Definição 7.1.23. *Uma sequência é dita* **sequência de Cauchy** *se, dado $\mathcal{E} > 0$, existe um natural n_0 tal que*

$$\|x_n - x_m\| < \mathcal{E}$$

para $n, m \geqslant n_0$.

Definição 7.1.24. *Um espaço métrico X é dito* **completo** *se todas as sequências de Cauchy tendem para um limite dentro do espaço.*

Definição 7.1.25. *Uma* **norma** *em X é uma função $\|\cdot\|\colon X \times X \Rightarrow \mathbb{R}$ onde valem as seguintes propriedades:*

i. $\|u\| \geqslant 0$. $\|u\| = 0 \Leftrightarrow u = 0$

ii. $\|\alpha \cdot u\| = |\alpha| \cdot \|u\|$

iii.$\|u + v\| \leqslant \|u\| + \|v\|$ *(desigualdade triangular)*

Exemplo 7.1.26. *Seja o vetor $u \in X$ tal que $u = (\xi_1, \xi_2, \cdots, \xi_n)$. ($\xi_j$) são os coeficientes de u na base $\{x_j\}$). Algumas normas conhecidas são:*

Norma 1. $\| \cdot \|_1$*:*

$$\|u\|_1 = \sum_{j=1}^{n} |\xi_j|$$

Norma 2. $\| \cdot \|_2$*:*

$$\|u\|_2 = \left(\sum_{j=1}^{n} |\xi_j|^2 \right)^{\frac{1}{2}}$$

Norma p. $\| \cdot \|_p$*:*

$$\|u\|_p = \left(\sum_{j=1}^{n} |\xi_j|^p \right)^{\frac{1}{p}}$$

Norma infinito. $\| \cdot \|_\infty$*, quando $p \to \infty$:*

$$\|u\|_\infty = \max\{|\xi_j|\}$$

Definição 7.1.27. *Seja $f \colon D \to \mathbb{R}$ diferenciável em $x \in D$. Se $f(x)$ satisfaz a condição necessária de primeira ordem, x é chamado de* **ponto estacionário**.

Definição 7.1.28. *Definimos a função Lagrangiana do seguinte modo:*

$$\mathcal{L}(x, \lambda) = f(x) - \sum (\lambda_i \cdot c_i(x))$$

onde $c_i(x)$ são as restrições de igualdade do problema e λ_i são os multiplicadores de Lagrange associados às restrições.

Não colocar

Definição 7.1.29. *Dizemos que um problema satisfaz as* **condições de otimalidade de KKT** *se existe* λ *que satisfaz:*

1. *O conjunto do gradiente das restrições ativas é L.I.*

2. $\mathcal{L}'(x, \lambda) = 0$

3. $c_i(x) = 0$ *se* c_i *é condição de igualdade.*

4. $c_j(x) \geq 0$ *se* c_j *é condição de desigualdade.*

5. $\lambda_i \geq 0$

6. $\lambda_i \cdot c_i(x) = 0$ *e* $\lambda_i \cdot c_i(x) = 0$

Definição 7.1.30. *Seja* $x^* \in \mathbb{R}^n$ *tal que* x^* *é minimizador local de*

$$min f(x), c_j(x) \geqslant 0, c_i(x) = 0$$

acrescentar numeração no problema restrito de minimização incluindo restrições de igualdade e desigualdade onde f, c_i *e* c_i *são continuamente diferenciáveis. Então existe um vetor* λ *com entradas* λ_i *que satisfaz as* **condições de otimalidade de KKT***:*

1. $\nabla_x \mathcal{L}(x^*, \lambda) = 0.$

2. $\lambda_i \geq 0$

3. $c_i(x) = 0$ *se* c_i *é condição de igualdade.*

4. $c_j(x) \geq 0$ *se* c_j *é condição de desigualdade.*

6. $\lambda_i \cdot c_i(x) = 0$ *e* $\lambda_i \cdot c_i(x) = 0$

Definição 7.1.31. *Dizemos que* $d_k \in \mathbb{R}^n$ *é uma* **direção de descida** *de* f *em* x^* *se* $f(x^* + d) \leqslant f(x)$.

Definição 7.1.32. *Seja $f\colon \mathbb{R}^n \to \mathbb{R}$. Dizemos que ∇f é o* **gradiente** *de f se as derivadas parciais existem e se:*

$$\nabla f(x) = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n}\right)^T$$

Definição 7.1.33. *Seja $f\colon \mathbb{R}^n \to \mathbb{R}$. Dizemos que $\nabla^2 f$ é a matriz* **hessiana** *de f se cada uma das derivadas parciais existem e se:*

$$\nabla^2 f(x) = \begin{bmatrix} \dfrac{\partial^2 f}{\partial x_1^2} & \dfrac{\partial^2 f}{\partial x_1\,\partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_1\,\partial x_n} \\ \dfrac{\partial^2 f}{\partial x_2\,\partial x_1} & \dfrac{\partial^2 f}{\partial x_2^2} & \cdots & \dfrac{\partial^2 f}{\partial x_2\,\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 f}{\partial x_n\,\partial x_1} & \dfrac{\partial^2 f}{\partial x_n\,\partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

Definição 7.1.34. *Uma matriz é dita* **quadrada** *quando o seu número de linhas é igual ao de colunas.*

Definição 7.1.35. *Seja B uma matriz quadrada dada por $B = b_{i,j}$. Dizemos que B^T é a matriz transposta de B se:*

$$B^T = b_{j,i}$$

Definição 7.1.36. *Seja B uma matriz quadrada. Dizemos que B é* **simétrica** *se $B = B^T$*

Definição 7.1.37. *Seja B uma matriz quadrada. Dizemos que B é* **não singular** *se existir uma matriz A tal que*

$$B * A = A * B = 1$$

se A não existe, dizemos que B é **singular**.

Bibliografia

[1] NOCEDAL, J.; WRIGHT, S., Numerical Optimization. 2nd ed. ed. [S.l.]: Springer, 2006.(Springer series in operations research) STEWART, J. Calculo. 7. ed. [S.l.]: Cengage, 2013. v. 1. Citado na página

[2] IZMAILOV, A.; SOLODOV, M. Otimização - volume 1 - Condições de Otimalidade, Elementos de Análise Convexa e de Dualidade. [S.l.]: impa, 2005.

[3] IZMALOV, A.; SOLODOV, M. Otimização - volume 2 - Métodos Computacionais. [S.l.]: IMPA, 2007.

[4] BURDEN, R. L.; Faires, J. D., Análise Numérica, Editora Thomson Learning, São Paulo, 2003.

[5] Python Software Foundation. Python Language Reference, version 2.7. Available at http://www.python.org

[6] VOLTOLINI, L., Representação de problemas Hock-Shitkowski em linguagem Python. Disponível em https://github.com/lvoltolini/Hock-Shitkowski

[7] VOLTOLINI, L., Códigos para resolução de problemas Hock-Shitkowski em lagrangiana aumentada e penalidade. Disponível em https://github.com/lvoltolini/Thesis-Codes

[8] ROSA, Carlos Fabiano, Série de Taylor e Aplicações. Disponível em https://repositorio.ufsc.br/handle/123456789/105435